A Study of Earth Science Software Reuse Enablement Systems

James J. Marshall, Ryan Gerard, Shahin Samadi, Robert E. Wolfe
NASA Goddard Space Flight Center
Greenbelt, MD, USA
james.marshall@gsfc.nasa.gov, ryan.gerard@gsfc.nasa.gov, shahin.samadi@gsfc.nasa.gov, robert.e.wolfe@nasa.gov

***Abstract*—The reuse of software assets can be critical to the development of large-scale software projects where budget and reliability are paramount. Yet many of the benefits of software reuse are either not recognized or overlooked. The majority of software assets are not made available to peers or a wider community. Therefore, a key activity in promoting software reuse is the initiative to increase the awareness of reuse enablement systems.**

An ideal Earth science reuse enablement system should ensure that reusable software assets are readily available to the software developers who want to use them to build new or enhance existing Earth science applications. This can be done by placing the assets into a software catalog or repository system. Many such systems exist, mostly outside of the Earth science domain, each designed for a particular purpose. Some are domain-specific, covering one particular subject area, while others are more general, covering a large variety of fields. Each is well-designed for its target audience. Many candidate systems that exist provide open-source software solutions. However, user goals, quality control, and overall usability determine the usefulness of a system to the community of Earth science software developers.

The software used to create these systems also varies, ranging from standard HTML to full repository software packages like Repository in a Box (RIB), which uses the Basic Interoperability Data Model (BIDM), IEEE Standard 1420.1. The NASA Earth Science Data System (ESDS) Software Reuse Working Group has examined a variety of these systems, and focused on their applicability to the Earth science domain. Within a set of requirements designed for the Earth science community, this paper compares selected features of these systems, such as providing reviews for assets or the software used to design the site, and how the presence or absence of these features affects the system's ability to promote reuse.

Software reuse; repository systems; Earth science; SEEDS; NASA

I. Introduction

Software reuse is the reapplication of a variety of kinds of knowledge about one system to another system in order to reduce the effort of developing and maintaining that system. There are many expected benefits of reuse, such as reducing cost, saving time, and increasing reliability [1, 2]. Productivity and quality improvements are typical motivations for reuse [3]. Productivity is often measured in terms of cost and labor, and reuse has the potential to decrease both, thereby increasing productivity. Increased productivity can also be used to reduce the time needed to start using the software, an important factor in the competitive research environment. Reusing software can also improve the reliability and quality of new products because the currently existing software components have already been tested and confirmed to perform according to their designs.

However, it can be difficult to locate suitable software to reuse. Surveys conducted by the NASA Earth Science Data System (ESDS) Software Reuse Working Group have shown that common reasons people do not reuse software is because they are unable to locate such software or did not know it existed [4]. Creating a catalog or repository in which to store reusable assets is useful, but that alone will not guarantee the systematic reuse of such assets [2].

The Software Reuse Working Group is chartered to oversee the development of a process that will maximize the reuse potential of existing software components while recommending strategies for maximizing the reusability potential of yet-to-be-designed components. As part of this work, we undertook a study of existing software catalog and repository systems to determine their role in serving the Earth science community and meeting reuse needs. In 2004, the Working Group collaborated for several months to select a set of important requirements needed for a system to meet the reuse needs of the community of Earth science software developers. A variety of existing systems were evaluated, as-is, against these requirements. Here, we describe the types of systems reviewed and note their performance in selected areas deemed important by the Working Group.

II. Description of Existing Systems

We examined a variety of existing software catalog and repository systems, including NASA and non-NASA sites in the study. A total of seven NASA sites and ten non-NASA sites were reviewed, and an additional nine sites were inspected, but not reviewed in detail. This was because, in general, they were too narrowly focused to meet the needs of the community of Earth science software developers.

The systems can be grouped according to their domain. We located domain-specific sites that are designed to meet the needs of a specific target community and field of study, for example, Earth science, astronomy and space science, or

This work was sponsored in part by the NASA/GSFC Earth Science Data Systems Working Group.

mathematics. Other sites are designed for the general science, technical, or engineering communities, containing software from a variety of specific fields within these more general areas. There are also sites with essentially no specific domain, ones that contain software in general, from any field or subject area; SourceForge is one example of such a site. Also, sites that only contain software written in one particular programming language typically have no specific domain.

We also found that the sites were generally designed for different audiences and with different purposes. For example, some sites focus on providing software and services in support of the data products they offer. This is certainly a useful service, but normally there are few, if any, reusable software assets that a software developer may find useful at such sites. Other sites have some restriction(s) on the types of software listed. These sites may contain some useful reusable software assets, but the restrictions limit software developers to a subset of all available assets. Therefore, users may need to look at other sites to find the assets they desire. Sites that are focused on one particular computer programming language typically target software developers. However, while these sites can be useful in locating reusable software assets, the lack of a specific domain decreases the chances of finding assets relevant to any one specific field.

III. Comparison of Features in Existing Systems

The Software Reuse Working Group spent several months developing a set of important requirements necessary for a software catalog/repository system to meet in order to meet the needs of the Earth science software developer community. These fell into a number of areas including general, asset usage, asset submission, content management, and system administration, and each area contained a number of other, more specific requirements. As it was not possible to examine most of the back-end features of the different systems, our examination took the approach of a user, rather than someone like an administrator, and so we primarily considered front-end features. We will discuss a few of the features included in our requirements, considering how well the underlying software of existing systems is able to implement the features and meet our requirements.

A. Reviews and Ratings

One important feature of an ideal software reuse catalog or repository system is the ability for users to provide reviews and/or ratings of assets in the system and to see the reviews and/or ratings provided by others. Reviews refer to text written by users. Ratings refer to a simple scoring mechanism, such as a 1-5 scale where users select one value to indicate their opinion of how good the asset is overall, and the average value of all ratings is displayed. Individual ratings may also be displayed. These reviews and ratings serve as a form of peer review for the assets in the catalog or repository system. Ratings will provide other users with a quick, general feel for how useful the asset is overall, while reviews provide more detailed information. Reviews may include descriptions of bugs or other problems (e.g., installation difficulties), areas where the asset is particularly useful, or how easy or difficult the asset was to reuse. All of this information can be used by users of the system to locate the asset that is most suitable for meeting their needs. In addition, it can provide the system administrators and managers with information useful in maintaining and cleaning up the system. For example, assets that consistently receive poor ratings and reviews should be removed from the system, or at least not made publicly available, in order to make it easier for users to locate assets that are more useful.

Systems that use, or at least produce, simple HTML web pages generally do not have the ability to accept user reviews and ratings. This may be because these features are difficult to code effectively, or because the systems using HTML pages did not deem reviews and ratings important enough to include. However, such web pages are typically static in nature and therefore do not allow basically real-time updating of the information contained in them. This can cause a delay in providing users with the most recent information, which can affect their decision on which assets to use.

The Repository in a Box (RIB) software package does not contain rating or review features. It was designed to create web-based metadata repositories, and it performs that task well. The back-end uses the Basic Interoperability Data Model (BIDM), IEEE Standard 1420.1, to store information. However, the web pages RIB generates are fairly simple and do not have the same kind of front-end enhancements as some other systems.

Some form of a review feature has been provided by systems that make use of a scripting language such as PHP to generate their web pages. However, we did not find ratings at these sites, and the review feature tended to be very simple. Again, this could be due to coding difficulty or decisions on what is important, but making use of a scripting language is beneficial for this feature. Software designed more for content management purposes also tends to provide review and rating features, if not built-in, then via modules that can be added to the base package. None of the sites we examined in detail appeared to use Content Management Software as their primary implementation, suggesting that while it may contain useful features, overall it may not be the best option for producing software asset catalogs and repositories.

B. Contributing and Updating Assets

A repository or catalog system needs to be populated with assets in order to be useful. As part of the creation of the system, an initial population will be performed, most likely by locating assets in other systems that are suitable for inclusion in the new system. However, allowing users of the system to contribute and update their assets is the best way to provide new content and keep existing content updated. It enables members of the target audience to take a more active role in the reuse process by giving them an easy way to distribute their products. The ability to update assets is also needed since software assets are often revised, for example, to fix bugs, to optimize the code, or to add additional features. It is important that users who contribute assets to a system have a way of updating those assets so that other users will have access to the most recent versions.

There are some systems that do not allow users, in general, to contribute and update assets. These systems are typically limited to providing assets produced by the organization to which the system belongs. Therefore, outside users are normally not allowed to upload their own assets to the system. However, most systems do provide some method for users to contribute assets, and the level of functionality does not seem to depend highly on the underlying software. There are sites that appear to use or produce simple HTML pages with about the same capabilities as sites using XML or a scripting language to provide their contribute/update feature. The same scripting language can be used to produce a feature with relatively low or high functionality.

As noted in the previous section, RIB is typically much stronger on the back-end than the front-end. The sites we found that used RIB had no obvious way for users to upload their own assets to the systems. It appears that this is not an inherent feature of RIB-created systems. However, since a variety of systems and underlying software do allow users to contribute assets, it seems like repositories generated by RIB could be modified to provide this feature. Perhaps we were unable to locate a RIB site with this feature because the ones we found were all limited to providing only software assets produced by the organization to which the repository belonged.

C. Automatic Notifications

A system can save users time and effort by providing automatic notifications of updates to the system and assets in it. Without such a feature, users would have to perform frequent manual checks of the system instead. The system must know how to contact users and be able to save their preferences, so users must be able to register these pieces of information with the system. This can be done through a site registration and an account for the whole site, which can be used for other purposes, but this is not required. Automatic notifications could be set up independent of a site registration. Users should be able to select the type of notifications they wish to receive. These may include when new versions of an asset they use are uploaded to the system, when a new asset in a particular category is added, or system information such as a scheduled outages for maintenance purposes. Another possibility is for users who contributed assets to request notifications of when other users rate or review those assets, but we have not seen this in the systems we examined. Notification via e-mail is likely to be the most commonly used method, but other methods are possible (e.g., RSS feed) and may be used instead of or in addition to e-mail.

Standard HTML web pages alone do not provide automatic notifications. This kind of interactive feature is beyond the capability of static HTML. Server-side scripting can provide this ability, but it would not be visible from a user's perspective. Sites that make use of scripting languages are able to provide users with automatic notifications. The details of the feature vary between systems since each system normally has to produce its own implementation of the feature. Systems vary in what events can trigger automatic notifications and how users select the events for which they wish to receive notifications. It appears that notification of updates to individual assets is the most common, and perhaps the simplest to implement, but selection based on categories (often the same as the hierarchies under which assets are classified) has been used as well.

Again, RIB does not provide this function, at least not in the systems we examined. It may be possible to modify RIB to allow automatic notifications. Since its main purpose is to provide the back-end of the system, modifications may be possible to improve the front-end characteristics of the web sites produced by RIB. We have not looked into this option.

D. Registering Asset Usage

One feature that very few sites provide is the ability to register asset usage. This is used to allow users to indicate that they are actively using an asset. This is not the same thing as downloading the asset. Downloads are commonly tracked and may be used as a measure of which assets are popular on a particular site. However, downloading an asset does not guarantee active usage of that asset. A user may download an asset, try it out, then decide that it does not fulfill his/her needs, and stop using it. This is why a separate indication of active usage is helpful in measuring the actual level of reuse and generating statistics about the reuse of the asset. This feature can also be used in a manner similar to automatic notifications, to provide users with information about bug fixes, updates, and enhancements to the asset, for example.

As noted, this feature was rarely seen in the sites we examined. The two sites that do offer it both utilize scripting languages in creating their web pages, and use a web-based form to capture the asset registration information. This simple method of registering assets could be implemented in other scripting languages, or a combination of standard HTML (to create the form) and a scripting language (to process the form). The RIB software package does not provide this feature by default, but it may be possible to modify the software, or the front-end interface to it, to provide asset registration. Since it appears that this is a relatively simple feature to implement, web-based forms are not that difficult to create and use, perhaps most systems did not deem asset registration important enough to implement.

IV. Conclusions

Our examination of different catalog and repository systems has shown that there are many different systems which are useful for their users. Each one provides a valuable service for its target audience and meets the needs of its audience. The software used in almost any of these systems could be reused, with appropriate modifications, to satisfy the needs of the community of Earth science software developers. Modifications would be necessary because existing systems have not targeted this audience, and additional features would be required to meet the needs of this audience.

Most systems are able to provide at least some functionality in most of our requirement areas. In general, the ones that provided the most functionality in the most areas made use of scripting languages. Some may have used catalog/repository software packages, possibly of their own design, but it seems clear that the abilities scripting languages possess are necessary to provide certain features. For example, sites that used, or at least generated, static HTML pages typically could not provide features such as automatic notifications or registering asset usage. These require more interaction with the user than static HTML can provide.

The Repository in a Box (RIB) software package appears to be a very useful system for generating a web-based repository of metadata. It is open source, and uses the Basic Interoperability Data Model (BIDM), IEEE Standard 1420.1. Other data models can be substituted for the BIDM if desired, so RIB provides very good control of how data are stored in the repository. However, it is very much a back-end system and the web pages it creates are fairly basic and static. Our viewpoint for this study was that of a user, looking primarily at the front-end of the system and how users interact with it. It appears that the RIB software or the web pages it creates can be modified to produce a more elaborate front-end with additional features. However, we have not examined this option in detail, so it is uncertain how much work would be required to do this.

The set of requirements we selected were deemed by the NASA Earth Science Data System (ESDS) Software Reuse Working Group to be the most important ones for a system serving the community of Earth science software developers. We have only covered a few of them here. Reviews and ratings help to encourage reuse by making it easier to determine which software assets are most suitable for each job. This is a form of peer review, and can also be used to help clean up the system periodically, removing (at least from public listings) assets which consistently receive poor ratings and reviews. The ability to contribute and update assets is a vital feature for any system intended to promote reuse. Without the ability for users to distribute their own products, reusable assets would not be available for others, and thus reuse would become a very difficult practice. Automatic notifications can provide users of a system with information about the newest versions of assets they already have, or direct them to completely new assets that they may find useful in their work. Providing users with updates like these can help encourage them to reuse existing assets when possible. Registering asset usage is more helpful in generating statistics regarding software reuse, but those statistics can in turn help encourage further reuse by showing the usefulness of reuse and how prevalent it is.

The more features a system possesses that help encourage the reuse of existing software assets, the better it is at enabling reuse among its users. Many systems provide their audiences with some features that encourage and enable software reuse, and these can be used as building blocks for future work. The software these systems use can serve as the basis for additional work to develop a catalog or repository of reusable software assets for the community of Earth science software developers. This is one of the issues that our Software Reuse Working Group is currently working on, and hopes to provide some solution for in the near future.

ACKNOWLEDGMENT

We thank the members of the ESDS Software Reuse Working Group for their contributions to this work.

REFERENCES

[1] R. E. Grinter, “From local to global coordination: lessons from software reuse,” Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work, pp. 144-153, 2001.

[2] J. Long, “Software Reuse Antipatterns”, ACM SIGSOFT Software Engineering Notes, vol. 26, iss. 4, pp. 68-76, July 2001.

[3] S. Samadi, N. Alameh, R. Wolfe, S. Olding, D. Isaac, “Strategies for Enabling Software Reuse within the Earh Science Community”, Proceedings of the 2004 IEEE International Geoscience and Remote Sensing Symposium, vol. 3, pp. 2196-2199, September 2004.

[4] J. J. Marshall, S. W. Olding, R. E. Wolfe, V. E. Delnore, “Software Reuse Within the Earth Science Community”, Proceedings of the 2006 IEEE International Geoscience and Remote Sensing Symposium, in press.